

CLARAty Estimation Architecture

Dan Gaines

Stergios Roumeliotis Issa Nesnas

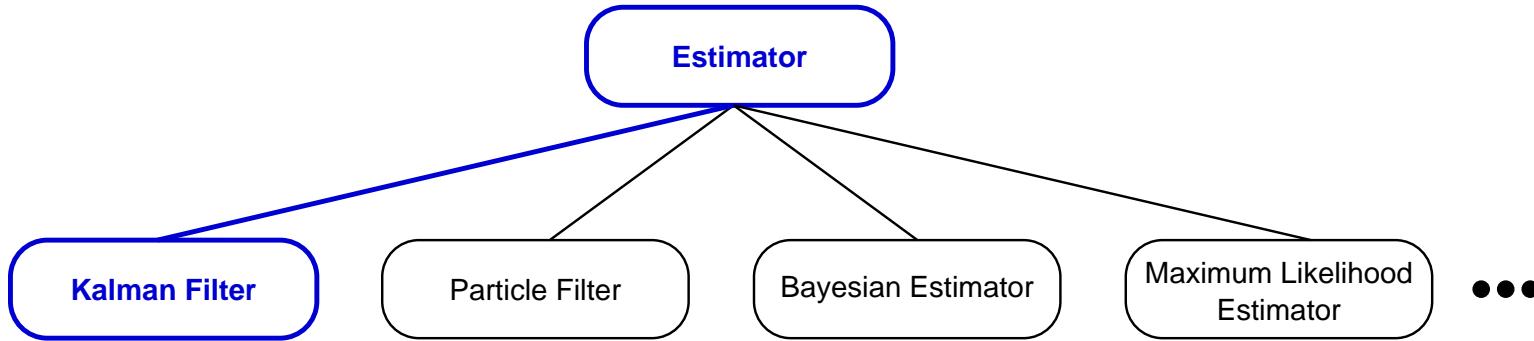
October 23, 2003

Overview of Estimation



- **Purpose:** Estimate the state of the system
- **Information:** Estimator may use one or more of:
 - system model: predicts next state
 - measurements: provides evidence about state
 - control: input into the system
- **Examples:**
 - position
 - velocity
 - feature / landmark position
 - temperature

Goals of CLARAty Estimation Architecture

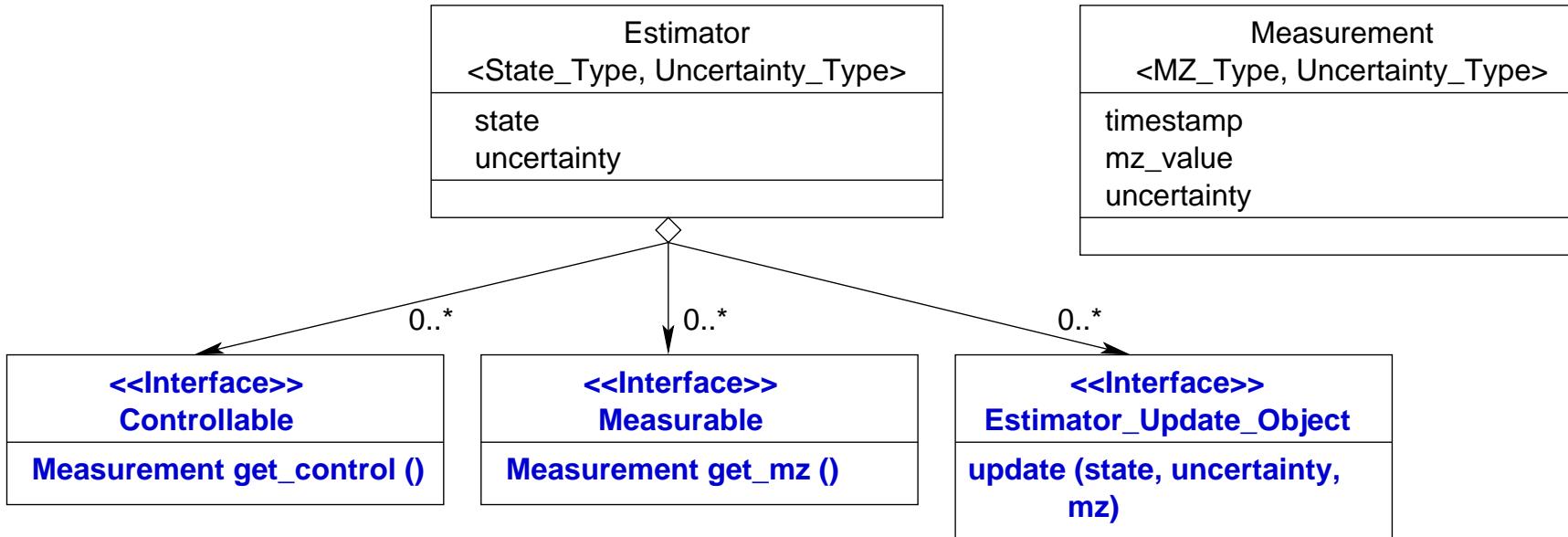


- Facilitate development of estimators
 - infrastructure common to estimators
 - infrastructure to support Kalman Filters
 - support for periodic computations
- Facilitate reuse of estimators
 - treat estimators as virtual sensors
 - common API enables swapping estimators
 - adaptable to other CLARAty platforms

Outline of Talk

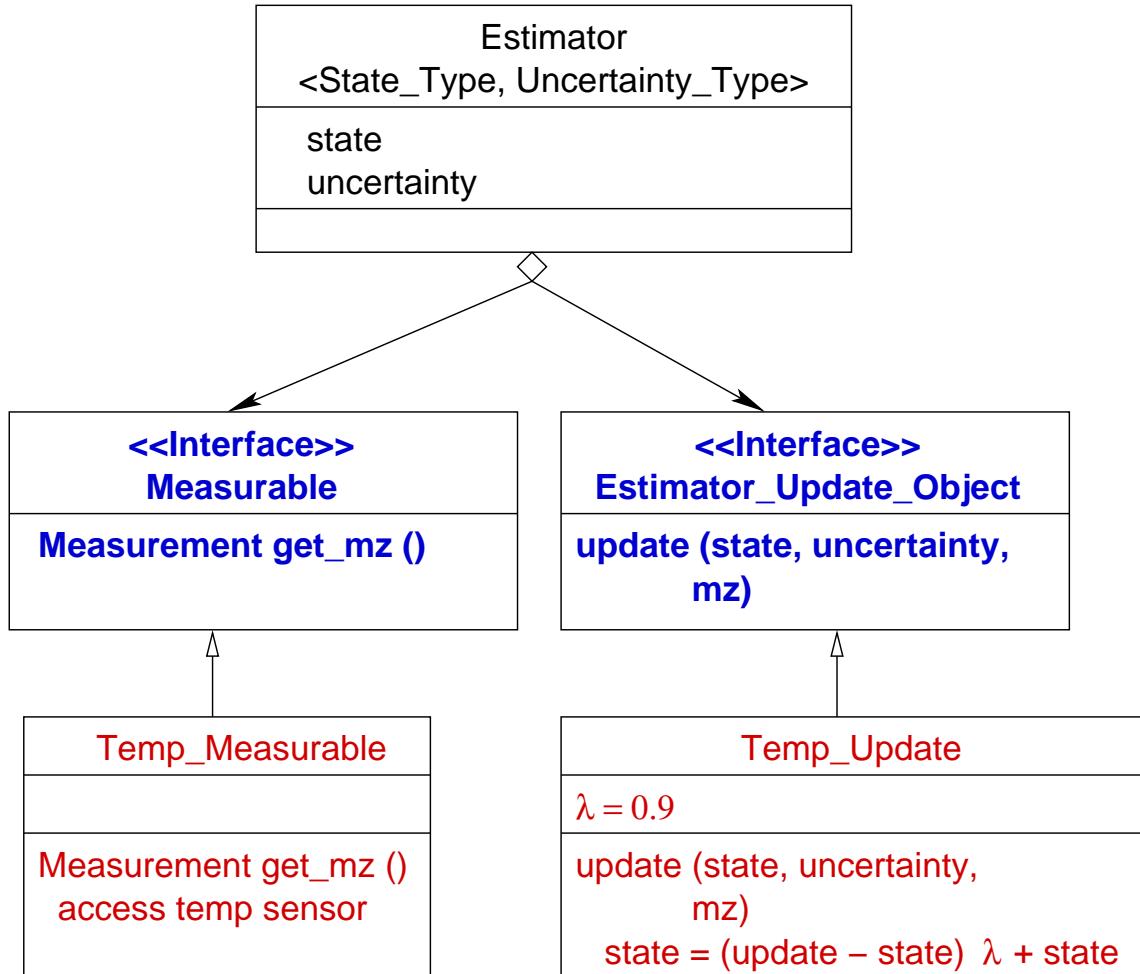
- Introduction
- Generic Estimators
- Kalman Filter Estimators
- Case Study - Fido 3DOF EKF
- Future Work - 6DOF EKF
- Conclusion

Generic Estimator

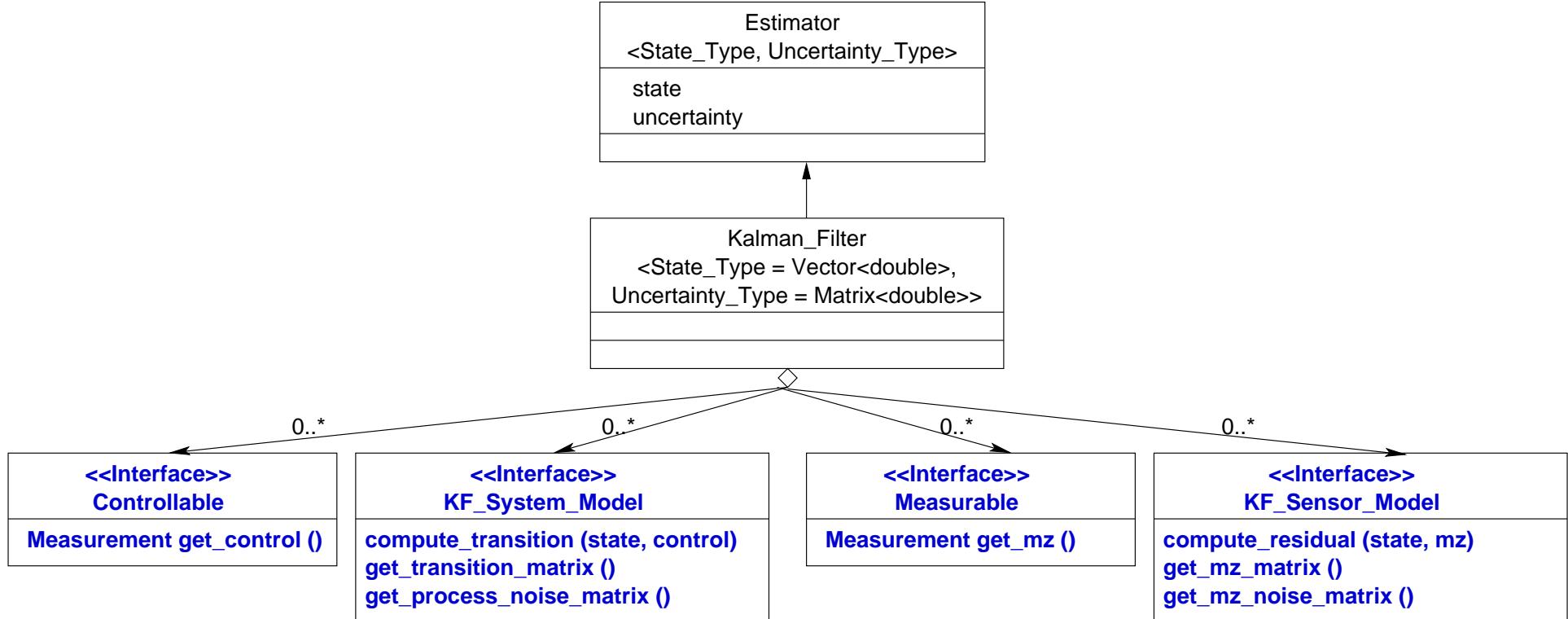


- Few assumptions made about how estimation is performed
- Actual estimators created by implementing interfaces
- Controllable Interface
 - acquire/translate control information
- Measurable Interface
 - acquire/translate sensor reading
- Estimator Update Interface
 - responsible for updating state and uncertainty
 - uses data from Measurable or Controllable
 - executed at user-specified frequency

Example Estimator – Estimating Temperature

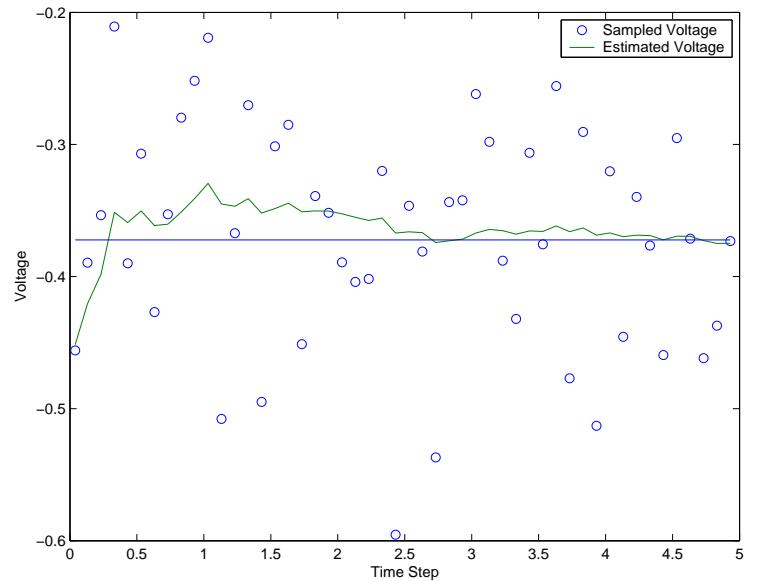
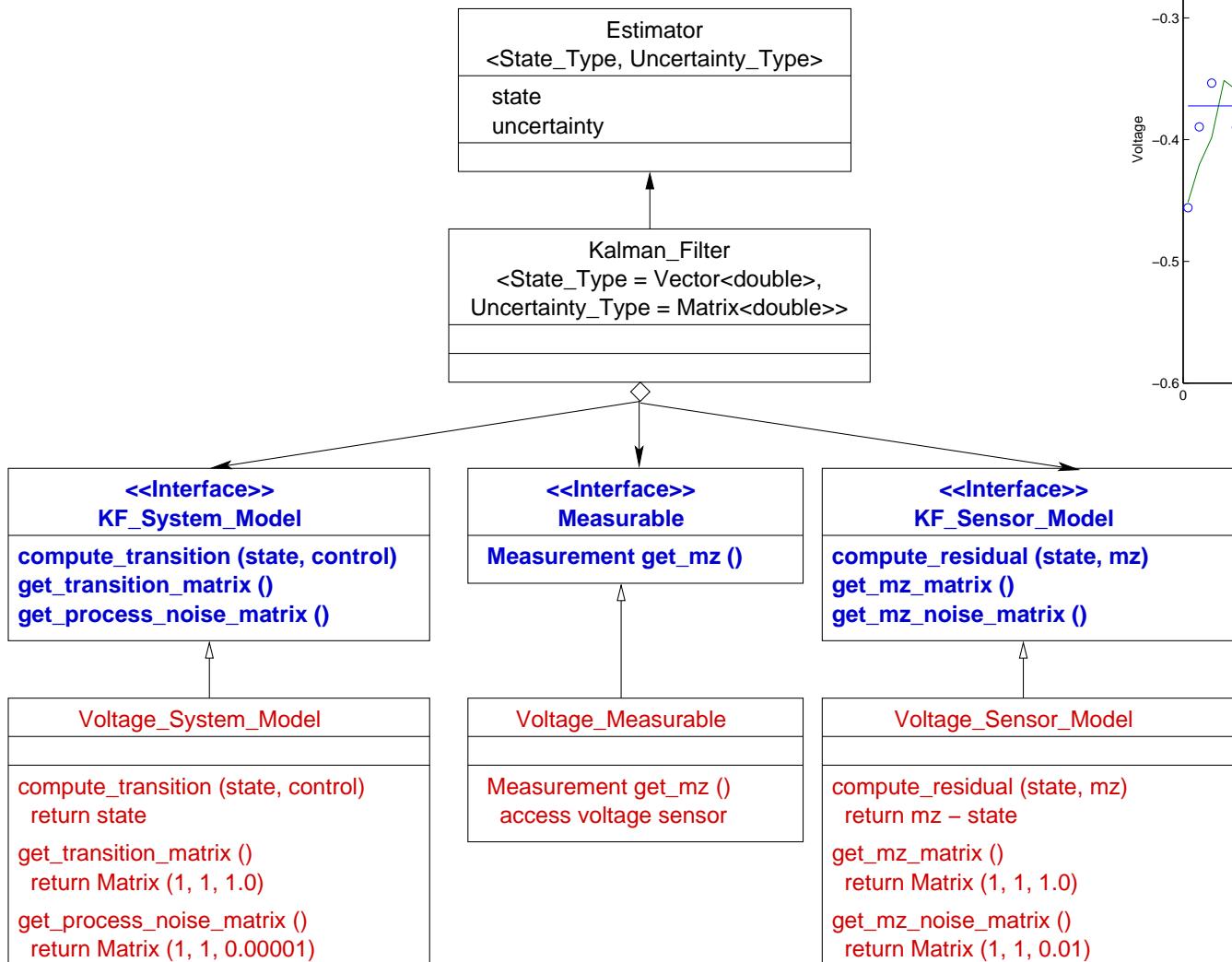


Kalman Filter



- System Model interface:
 - performs state transition
 - provides system noise information
- Sensor Model interface
 - make sensor prediction
 - provide sensor noise information
- Runs each model at user-specified frequency

Example Kalman Filter – Estimating Voltage¹



¹ Adapted from “An Introduction to the Kalman Filter”, Welch & Bishop, UNC Chapel Hill, 2003.

Overview of FIDO EKF Algorithm²

Role: Provide 3DOF pose estimation for rover (x, y heading)

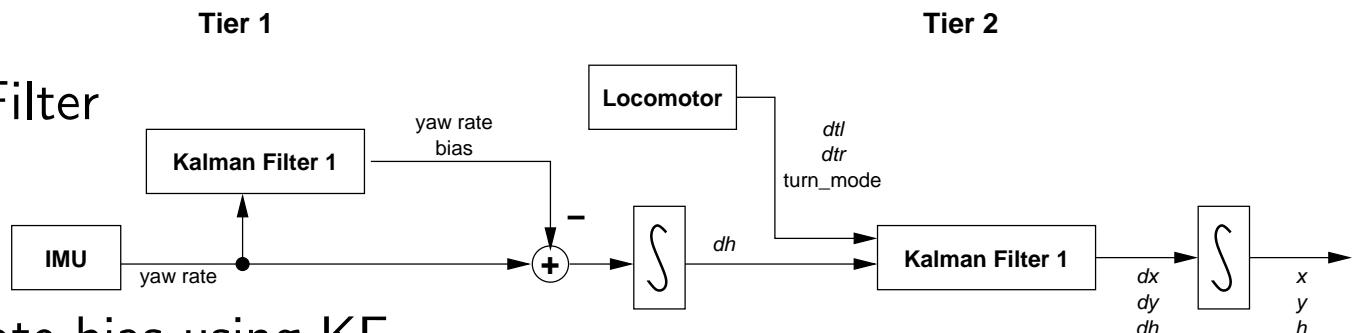
Input:

- drive command
- wheel encoders
- Inertial Measuring Unit (IMU)
 - currently using only one gyro (yaw rate)

Design: Two-tier Kalman Filter

Tier 1: while stationary

- uses IMU yaw rate
- estimate IMU yaw rate bias using KF



Tier 2: while moving

- uses corrected IMU yaw rate and wheel encoders
- fuse IMU yaw rate and encoder estimations within KF
- estimate change in rover x, y and heading
- integrate x, y and heading to keep track of 3DOF pose

Assumption: rover moves in short increments so bias estimate is valid during move

²Algorithm developed by Eric Baumgartner.

KF Tier 1: Estimating Yaw Rate Bias

- Linear Kalman Filter
 - runs when rover is stationary
- System model when rover is stationary: next state (i.e. bias) is same as previous state

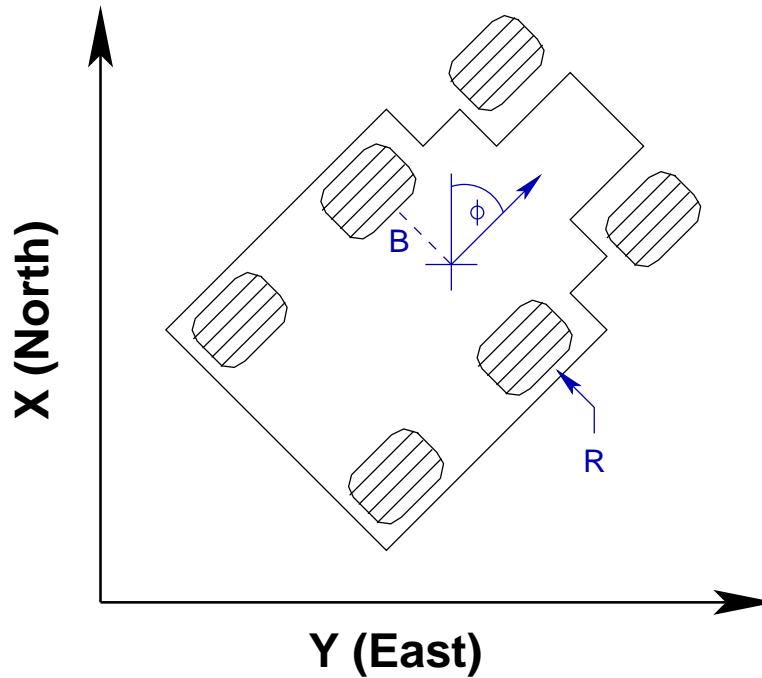
$$\frac{dbias(t)}{dt} = 0 + w_{bias} \text{(process noise)}$$

- Sensor model: if yaw rate bias is b , expect yaw rate measurement b

$$z_{bias} = \text{bias} + u_{bias} \text{(measurement noise)}$$

- **Assumption:** rover moves in short increments so the bias estimate is valid during move

Vehicle Kinematic Model



- Define virtual middle wheel with parameters:

$$\begin{aligned}\alpha &= \frac{d\theta_l + d\theta_r}{2} \quad \text{velocity} \\ u &= \frac{d\theta_l - d\theta_r}{d\theta_l + d\theta_r} \quad \text{angle}\end{aligned}$$

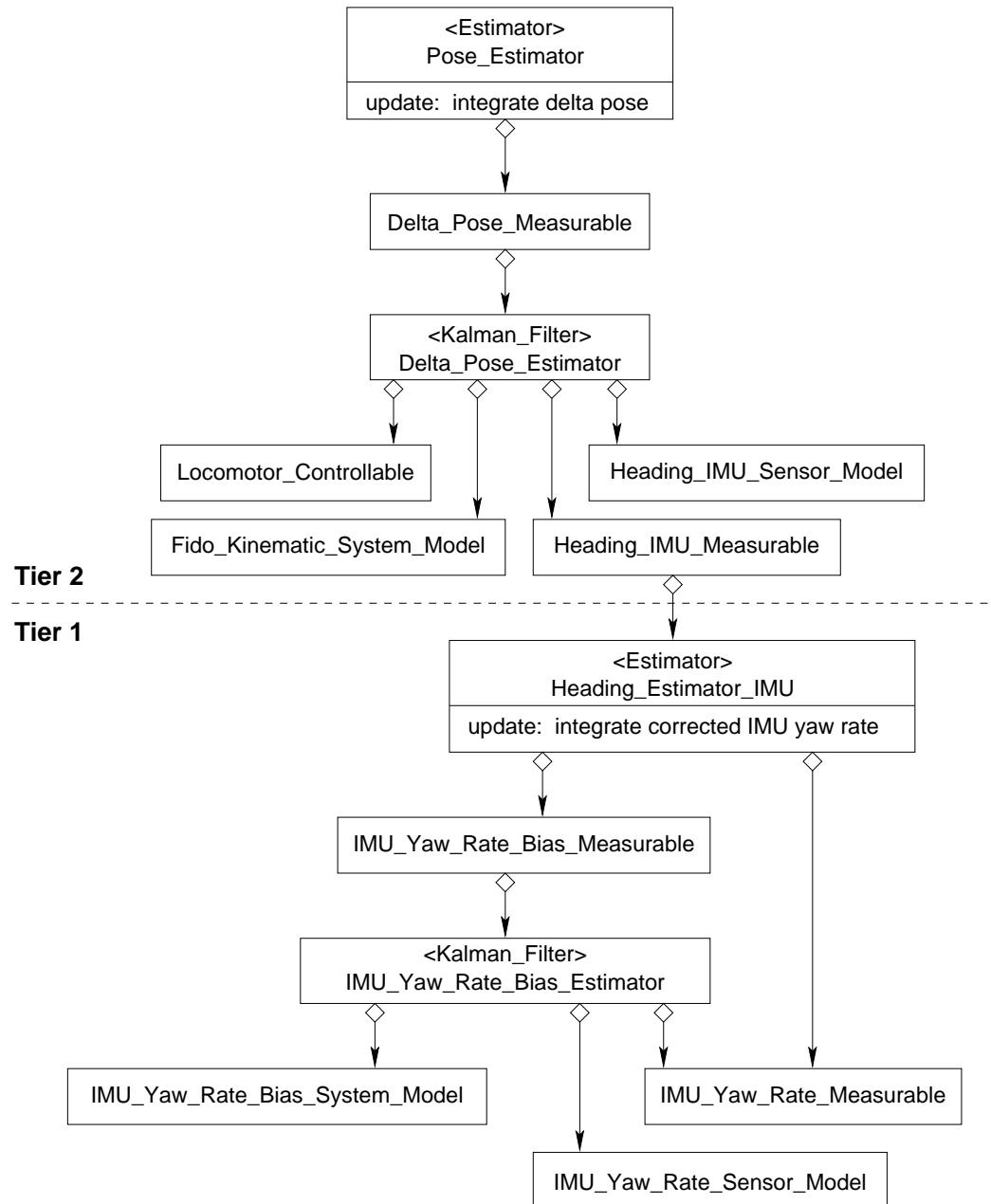
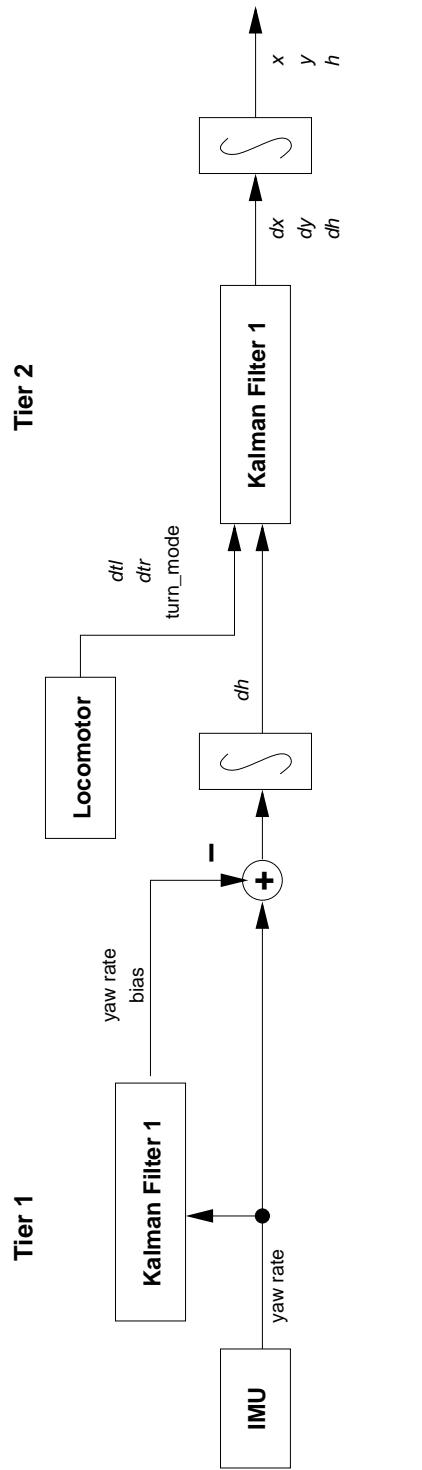
- State equations:

$$\frac{d\mathbf{x}(\alpha)}{d\alpha} = \begin{bmatrix} \frac{dX(\alpha)}{d\alpha} \\ \frac{dY(\alpha)}{d\alpha} \\ \frac{d\phi(\alpha)}{d\alpha} \end{bmatrix} = \begin{bmatrix} R(\cos\phi)\alpha \\ R(\sin\phi)\alpha \\ \frac{R}{B}u\alpha \end{bmatrix}$$

KF Tier 2: Estimating Delta X, Y and Heading

- Extended Kalman Filter
 - runs when rover is moving
- System model: see previous slides
- Sensor model: relates rover pose to IMU relative heading
 - if delta heading is θ , expect IMU relative heading to be θ

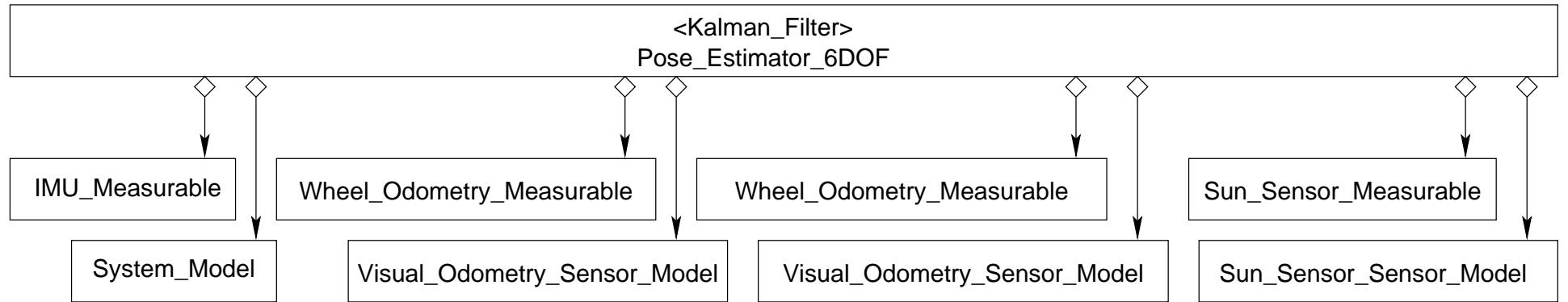
Implementation in CLARAty Estimation Architecture



Fido EKF Implementation Notes

- Initial implementation with Rocky 8 platform – about 3 months
 - including implementing estimation architecture
- Adaptation to Fido platform – 1 day
- Adaptation to Roams platform – about 2.5 weeks
 - including design/implementation of CLARAty/Roams interface

Future Work – 6DOF EKF



Conclusions

Contributions:

- support for developing new estimators within CLARAty
- facilitates adaptation to new platforms
- facilitates swapping estimators on a given platform

Future Work:

- extend the set of estimators in CLARAty
- add support for other estimators